1. (Currently amended) A method for mapping managed object metadata, the method comprising:

receiving a plurality of communications each pertaining to a different one of a plurality of managed objects from a management server for the managed objects, wherein each communication comprises data typed according to an abstract syntax notation; and

accessing a converter interface for each communication for converting the abstract syntax notation data types of each communication to interface definition language data types, wherein the same converter interface is accessed for each of the managed objects such that said converting is generic to the managed objects, wherein said converting comprises:

inputting a first data type from a first set of data types, wherein the first set of data types is expressed in ~~an~~ the abstract syntax notation, and wherein the abstract syntax notation comprises a language for describing data;

determining a corresponding second data type from a second set of data types, wherein the second set of data types is expressed in ~~an~~ the interface definition language, wherein the interface definition language comprises a language for implementing interfaces to managed objects, wherein the interface definition language is operable across a plurality of platforms and across a plurality of programming languages, and wherein the interface definition language is class independent; and

returning the second data type.

2. (Original) The method of claim 1, wherein the managed object metadata comprises metadata concerning a telephone system.

3. (Original) The method of claim 1, wherein the managed object metadata comprises metadata concerning a network switch.

4. (Original) The method of claim 1, wherein the metadata comprises type information about an attribute of one of the managed objects.

5. (Original) The method of claim 1, wherein the metadata comprises type information about an action of one of the managed objects.

6. (Original) The method of claim 1, wherein the metadata comprises type information about a notification of one of the managed objects.

7. (Original) The method of claim 1, wherein the first data type comprises a primitive data type in the abstract syntax notation, and wherein the second data type comprises a generic primitive data type in the interface definition language.

8. (Previously presented) The method of claim 1, wherein the first data type comprises an object data type in the abstract syntax notation, and wherein the second data type comprises a sequence of a generic primitive data type in the interface definition language.

9. (Original) The method of claim 1, wherein the first data type comprises an object data type in the abstract syntax notation, wherein the second data type comprises a choice structure of the interface definition language, and wherein the choice structure comprises:

a generic value field;

a plurality of data types; and

a selector, wherein the selector is an index whose value determines which of the plurality of data types is used to interpret the value in the generic value field.

10. (Original) The method of claim 1, wherein the abstract syntax notation comprises Abstract Syntax Notation One.

11. (Original) The method of claim 1, wherein the abstract syntax notation comprises Guidelines for Managed Objects (GDMO).

12. (Previously presented) The method of claim 1, wherein the interface definition language is operable to provide a mapping which is applicable to any managed object class.

13. (Currently amended) A method for mapping managed object metadata, the method comprising:

receiving a plurality of communications each pertaining to a different one of a plurality of managed objects from a manager for the managed objects, wherein each communication comprises data typed according to an interface definition language;

accessing a converter interface for each communication for converting the interface definition language data types of each communication to abstract syntax notation data types, wherein the same converter interface is accessed for each of the managed objects such that said converting is generic to the managed objects, wherein said converting comprises:

inputting a first data type from a first set of data types, wherein the first set of data types is expressed in ~~an~~ the interface definition language, wherein the interface definition language comprises a language for implementing interfaces to managed objects, wherein the interface definition language is operable across a plurality of platforms and across a plurality of programming languages, and wherein the interface definition language is operable to provide a mapping which is applicable to any managed object class;

determining a corresponding second data type from a second set of data types, wherein the second set of data types is expressed in ~~an~~ the abstract syntax notation, and wherein the abstract syntax notation comprises a language for describing data; and

returning the second data type.

14. (Original) The method of claim 13, wherein the managed object metadata comprises metadata concerning a telephone system.

15. (Original) The method of claim 13, wherein the managed object metadata comprises metadata concerning a network switch.

16. (Original) The method of claim 13, wherein the metadata comprises type information about an attribute of one of the managed objects.

17. (Original) The method of claim 13, wherein the metadata comprises type information about an action of one of the managed objects.

18. (Original) The method of claim 13, wherein the metadata comprises type information about a notification of one of the managed objects.

19. (Original) The method of claim 13, wherein the first data type comprises a generic primitive data type in the interface definition language, and wherein the second data type comprises a primitive data type in the abstract syntax notation.

20. (Previously presented) The method of claim 13, wherein the first data type comprises a sequence of a generic primitive data type in the interface definition language, and wherein the second data type comprises an object data type in the abstract syntax notation.

21. (Original) The method of claim 13, wherein the first data type comprises a choice structure of the interface definition language, wherein the choice structure comprises:

a generic value field;

a plurality of data types; and

a selector, wherein the selector is an index whose value determines which of the plurality of data types is used to interpret the value in the generic value field; and

wherein the second data type comprises an object data type in the abstract syntax notation.

22. (Original) The method of claim 13, wherein the abstract syntax notation comprises Abstract Syntax Notation One.

23. (Original) The method of claim 13, wherein the abstract syntax notation comprises Guidelines for Managed Objects (GDMO).

24. (Previously presented) The method of claim 13, wherein the interface definition language is operable to provide a mapping which is applicable to any managed object class.

25. (Currently amended) A computer system for mapping managed object metadata, wherein the system comprises:

a CPU;

a memory coupled to the CPU, wherein the memory stores program instructions executable by the CPU, and wherein the program instructions are executable to implement a generic converter for converting abstract syntax notation data types to interface definition language data types for metadata pertaining to a plurality of different managed objects, wherein the same converter interface is accessed for each of the managed objects, wherein the generic converter is configured to:

input a first data type from a first set of data types, wherein the first set of data types is expressed in ~~an~~ the abstract syntax notation, and wherein the abstract syntax notation comprises a language for describing object data;

determine a corresponding second data type from a second set of data types, wherein the second set of data types is expressed in ~~an~~ the interface definition language, wherein the interface definition language comprises a language for implementing interfaces to managed objects, wherein the interface definition language is operable across a plurality of platforms and across a plurality of programming languages, and wherein the data types in the interface definition language are class independent; and

return the second data type.

26. (Original) The computer system of claim 25, wherein the managed object metadata comprises metadata concerning a telephone system.

27. (Original) The computer system of claim 25, wherein the managed object metadata comprises metadata concerning a network switch.

28. (Original) The computer system of claim 25, wherein the abstract syntax notation comprises Abstract Syntax Notation One.

29. (Original) The computer system of claim 25, wherein the abstract syntax notation comprises Guidelines for Managed Objects (GDMO).

30. (Original) The computer system of claim 25, wherein in determining the corresponding second data type from the second set of data types, the program instructions are executable to look up the second data type in one or more lookup tables.

31. (Currently amended) A carrier medium comprising program instructions for mapping managed object metadata, wherein the program instructions are computer-executable to implement:

receiving a plurality of communications each pertaining to a different one of a plurality of managed objects from a management server for the managed objects, wherein each communication comprises data typed according to an abstract syntax notation;

accessing a converter interface for each communication for converting the abstract syntax notation data types of each communication to interface definition language data types, wherein the same converter interface is

accessed for each of the managed objects such that said converting is generic to the managed objects, wherein said converting comprises:

inputting a first data type from a first set of data types, wherein the first set of data types is expressed in ~~an~~ the abstract syntax notation, and wherein the abstract syntax notation comprises a language for describing data;

determining a corresponding second data type from a second set of data types, wherein the second set of data types is expressed in ~~an~~ the interface definition language, wherein the interface definition language comprises a language for implementing interfaces to managed objects, wherein the interface definition language is operable across a plurality of platforms and across a plurality of programming languages, and wherein the interface definition language is class independent; and

returning the second data type.

32. (Previously presented) The carrier medium of claim 31, wherein the first data type comprises an object data type in the abstract syntax notation, and wherein the second data type comprises a sequence of a generic primitive data type in the interface definition language.

33. (Currently amended) A carrier medium comprising program instructions for mapping managed object metadata, wherein the program instructions are computer-executable to implement:

receiving a plurality of communications each pertaining to a different one of a plurality of managed objects from a manager for the managed objects,

wherein each communication comprises data typed according to an interface definition language;

accessing a converter interface for each communication for converting the interface definition language data types of each communication to abstract syntax notation data types, wherein the same converter interface is accessed for each of the managed objects such that said converting is generic to the managed objects, wherein said converting comprises:

inputting a first data type from a first set of data types, wherein the first set of data types is expressed in ~~an~~ the interface definition language, wherein the interface definition language comprises a language for implementing interfaces to managed objects, wherein the interface definition language is operable across a plurality of platforms and across a plurality of programming languages, and wherein the interface definition language is operable to provide a mapping which is applicable to any managed object class;

determining a corresponding second data type from a second set of data types, wherein the second set of data types is expressed in ~~an~~ the abstract syntax notation, and wherein the abstract syntax notation comprises a language for describing data; and

returning the second data type.

34. (Previously presented) The carrier medium of claim 33, wherein the first data type comprises a sequence of a generic primitive data type in the interface definition language, and wherein the second data type comprises an object data type in the abstract syntax notation.